**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS**

**PUC Minas Virtual**

**Pós-graduação *Lato Sensu* em Engenharia de *Software***

Trabalho de Conclusão de Curso

Sistema de Gerenciamento de Projetos Vilicus Operis

Guilherme da Franca Batista

Belo Horizonte

Agosto, 2021

# Trabalho de Conclusão de Curso

**Sumário**

Trabalho de Conclusão de Curso 3

1. Cronograma de trabalho 4

2. Diagrama de casos de uso 5

3. Requisitos não-funcionais 5

4. Protótipo navegável do sistema 5

5. Diagrama de classes de domínio 6

6. Modelo de componentes 7

6.1. Padrão arquitetural 7

6.2. Diagrama de componentes 7

6.3. Descrição dos componentes 7

7. Diagrama de implantação 8

8. Plano de Testes 9

9. Estimativa de pontos de função 10

10. Informações da implementação 11

11. Referências 12

## Cronograma de trabalho

|  |  |  |  |
| --- | --- | --- | --- |
| **Datas** | | **Atividade / Tarefa** | **Produto / Resultado** |
| **De** | **Até** |
| 20/06/2021 | 22/06/2021 | 1. Análise de RF e RNF | Insumo para próximas etapas |
| 23/06/2021 | 25/06/2021 | 1. Modelagem de dados. | MER |
| 26/06/2021 | 28/06/2020 | 1. Desenho do diagrama de casos de uso do sistema. | Diagrama de casos de uso |
| 29/06/2021 | 01/07/2021 | 1. Análise de Requisitos Não Funcionais | Requisitos Não Funcionais |
| 02/07/2021 | 04/07/2021 | 1. Desenho do diagrama de classes. | Diagrama de classes de domínio |
| 05/07/2021 | 07/07/2021 | 6. Definição de padrão arquitetural. | Padrão Arquitetural |
| 08/07/2021 | 10/07/2021 | 7. Análise dos componentes. | Descrição dos componentes |
| 11/07/2021 | 13/07/2021 | 8. Desenho do diagrama de componentes do sistema. | Diagrama de componentes |
| 14/07/2021 | 16/07/2021 | 9. Desenho do diagrama de implantação do sistema. | Diagrama de implantação |
| 17/07/2021 | 18/07/2021 | 10. Desenho e definição do plano de testes do sistema. | Plano de testes |
| 19/07/2021 | 23/07/2021 | 11. Análise dos pontos de função. | Estimativa de pontos de função |
| 24/07/2021 | 25/07/2021 | 12. Análise dos componentes *backend*. | Informações da implementação |
| 26/07/2021 | 26/07/2021 | 13. Escrita de referências bibliográficas. | Referencia bibliografica |
| 27/07/2021 | 31/07/2021 | 14. Desenho de protótipo no *Balsamiq*. | Protótipo Navegável |
| 01/08/2021 | 03/08/2021 | 15. Implementação *POC* de *backend*. | Repositório *Backend* |
| 04/08/2021 | 06/08/2021 | 16. Implementação *POC* de *middleware*. | Repositório *Middleware* |
| 07/08/2021 | 09/08/2021 | 17. Gravação do vídeo sobre *wireframe*. | Vídeo |
| 10/08/2021 | 15/08/2021 | 18. Revisão e conclusão do trabalho. | Trabalho de conclusão |

## Diagrama de casos de uso

Diagram, schematic

Description automatically generated

Figura 1- Diagrama de casos de uso: <https://github.com/akademicmensch/puc-engenharia-de-software/blob/main/Diagramas/pngs/diagrama-de-casos-de-uso.png>

## Requisitos não-funcionais

Ao analisar o documento de requisitos de usuário propostos para o desenvolvimento do sistema de gerenciamento de projtos, foram extraídos os seguintes requisitos não-funcionais:

* **Segurança**: o sistema deve ter controle de segurança baseado em perfis de acesso.
* **Portabilidade**: o sistema deve ser acessado por dispositivos móveis, *tablets*.
* **Legal**: ainda que não esteja explícito no documento, deve-se levar em consideração a lei geral de proteção de dados pessoais, vigente desde o mês de agosto de 2020.
* **Usabilidade**: de modo geral, pois este requisito aparece em mais de duas descrições de requisitos funcionais, o sistema deve ser de fácil utilização.

Entendidos os requisitos não funcionais, estes foram decisivos para algumas decisões arquiteturais como o protocolo de autorização *OAuth 2.0*, visto que os usuários podem acessar o sistema de qualquer dispositivo, móvel ou imóvel, através da *Web*.

Além disso, a própria integridade física do sistema foi repensada. Decidiu-se por manter o padrão de microsserviços para o *backend* e um *frontend* separado com possibilidade de divisão com o crescimento do sistema e suas funcionalidades.

Estes requisitos não funcionais implicaram então na adoção do paradigma de microsserviços, que nos leva a uma arquitetura distribuída, *stateless* com a necessidade de autenticação e autorização centralizadas num único *entry point*.

## Protótipo navegável do sistema

O *link* direto para o diretório do reposítorio *GitHub* onde se encontra o *pdf* exportado do programa Balsamiq é este:

<https://github.com/akademicmensch/puc-engenharia-de-software/tree/main/Prototipo%20Navegavel/pdf>

Para realizar o download do pdf diretamente, use este link:

<https://github.com/akademicmensch/puc-engenharia-de-software/raw/main/Prototipo%20Navegavel/pdf/wireframe-vilicusOperis.pdf>

O *link* contendo o direcionamento para o vídeo do *YouTube* pode ser encontado no repositório:

<https://github.com/akademicmensch/puc-engenharia-de-software/blob/main/Prototipo%20Navegavel/video/link-para-video.txt>

## Diagrama de classes de domínio

## Diagram Description automatically generated

Figura - Diagrama de classes de domínio: <https://github.com/akademicmensch/puc-engenharia-de-software/blob/main/Diagramas/pngs/diagrama-de-classes.png>

## Modelo de componentes

### Padrão arquitetural

Após análise dos requisitos funcionais e não funcionais, decidiu-se por um modelo arquitetural híbrido. O *backend* do sistema proposto está baseado no paradigma de microsserviço e um frontend único, ainda que isto possa ser facilmente alterado para a ideia de microfrontends com a melhoria e avanço do sistema.

Além disso, alguns outros padrões foram adotados para facilitar a divisão de responsabilidades dos serviços e também a segurança de modo geral. O *backend* está classificado em duas categorias, *core* e *backend-for-frontend*(*BFF*). Os microsserviços sob a categoria core são os responsáveis por realizar operações transacionais com a base de dados. Comandos de *Data Manipulation Language*(*DML*) ficam restritos a estes serviços pois os mesmos não terão node ports expostas na *internet*, de modo que apenas um outro tipo de serviço possa acessá-lo, o *BFF*. Sob o domínio core estão os serviços *notification-core.app*, *report-core.app*, *project-core.app* e *authentication-core.app*.

Os *BFF’s* são os microsserviços com node ports expostos, i.e., serviços que podem ser acessíveis publicamente através de requisições vindas do *frontend*, passando necessariamente pelo gateway para verificação de autorização. Foram propostos dois serviços nessa categoria, o *adm-bff.app* que fica responsável por se comunicar com *notification-core.app* e *report-core.app* e o *project-bff.app* que fica responsável por se comunicar com o *project-core.app*. Esta divisão foi pensada por conta de um melhor controle relacionado aos domínios de negócio. O primeiro *BFF* fica responsavel por operações ligadas a sistemas externos(sistema de notificações e relatórios)e o segundo fica responsável pelas operações relacionadas diretamente ao negócio

Por fim, de acordo com a distribuição dos componentes com responsabilidades bem definidas, cria-se a possibilidade de substituição de tecnologia de acordo com a necessidade do projeto e evolução do sistema. *Frameworks* utilizados no *backend* e *frontend* podem ser subtituídos, *Gateway* e *Authorization Provider* podem ser substuídos por soluções mais recentes de mercado.

As tecnologias e frameworks utilizados no projeto são:

|  |  |
| --- | --- |
| *Java 8* | *CapacitorJs* |
| *Spring Boot* | *Angular 11* |
| *Docker* | *Kubernetes* |

No entanto, uma descrição mais detalhada será apresentada na seção 6.3(Descrição dos componentes).

### Diagrama de componentes

Diagram

Description automatically generated

Figura - Diagrama de componentes: <https://github.com/akademicmensch/puc-engenharia-de-software/blob/main/Diagramas/pngs/diagrama-de-componentes.png>

### Descrição dos componentes

Abaixo apresentaremos uma pequena descrição dos componentes com a tecnologia utilizada e padrões adotados.

1. **gateway.app** – Componente de *entrypoint* do sistema

O gateway do sistema, que oferece um *single entry point* para todos os endpoints expostos nos *BFF’s* é um projeto *Spring Boot* que utiliza como dependência principal o *Spring Cloud Starter Gateway*.

1. **discovery.app** – Componente de *discovery* para integração do *gateway*

Ao utilizar a *stack* de *Java* e *Spring Boot*, um componente intermediário de *discovery* faz-se necessário para que o *gateway* possa ter ciência dos serviços com os quais faz redirecionamentos. Além disso, este componente oferece aos admninstradores do sistema uma forma rápida e fácil de acompanhar a saúde dos microsserviços, indicando alguns *KPI’s*(*key performance indicators*) em uma tela de administração geral.

1. **authentication-core.app** – Componente de autenticação e autorização

Componente *Spring Boot* responsável por realizar a autenticação, autorização e administração geral dos usuários do sistema. Possui acesso exclusivo à base de usuários.

1. **project-core.app** – Componente principal de negócio

Componente *Spring Boot*, com acesso à base de projetos responsável por realizar operações *DML* relacionadas ao projeto e entidades auxiliares como tarefas, marcos e status.

1. **project-bff.app** – Componente exposto para consumo do frontend

Componente *Spring Boot* responsável por expor os recursos consumidos pelo frontend através do gateway. Os *node ports* do *pod* que contém o *project-bff.app* estão expostos para consumo.

1. **notification-core.app** – Componente-sistema de notificações

Componente *Spring Boot* responsável por realizar operações relacionadas às notificações do sistema. Seu domínio específico é a *TB\_NOTIFICACAO* onde estão parametrizados os tipos de notificações que o sistema envia por *e-mail* para os usuários. Diferente dos outros componentes do sistema, este componente funciona sem interação externa. De acordo com um intervalo de tempo pré definido, o componte é acionado e de acordo com os tipos de registros parametrizados, o componente monta as notificações e as envia, por *e-mail*, aos colabores e gerentes que necessitam ser avisados de atrasos em tarefas.

1. **report-core.app** – Componente-sistema de relatórios

Componente *Spring Boot* responsável por realizar operações relacionadas aos relatórios que os gerentes podem gerar. Seu domínio específico é a tabela *TB\_RELATORIO* onde estão parametrizados os tipos de relatórios, e onde podem ser adicionados novos quando necessário, que serão mostrados para o gerente requisitante.

1. **adm-bff.app** – Componente exposto para consumo do frontend

Componente *Spring Boot* responsável por expor as funcionalidades dos domínios *report-core.app* e *notification.core.app*. Assim como o *project-bff.app*, a existência deste componente está relacionada à segurança interna do componente, i.e., não expor serviços que acessam a base de dados.

1. **frontend.app** – Componente de frontend acessível via navegador

Componente *Angular* responsável por oferecer a interface gráfica do sistema para os usuários. A construção de telas ‘componetizadas’ deve ser planejada com cuidado pois dois dos requisitos não funcionais do sistema estão diretamente relacionados à visão gráfica do usuário final, a portabilidade e a usabilidade. Então, o componente deve possuir uma natureza responsiva em todas os seus detalhes.

1. **Banco de dados *RDS Mysql***  –

Base de dados *Mysql* em nuvem, provida pela *AWS*. O *schema* e a instância da base são únicos, a governança dos domínios estarão de baixo dos microsserviços responsáveis pela manutenção dos dados confiados a eles.

## Diagrama de implantação

Diagram, schematic

Description automatically generated

Figura - Diagrama de implantação: <https://github.com/akademicmensch/puc-engenharia-de-software/blob/main/Diagramas/pngs/diagrama-de-implantacao.png>

## Plano de Testes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Número** | **Caso de uso** | **Objetivo do caso de teste** | **Entradas** | **Resultados esperados** |
|  | *Login* na aplicação com controle de perfil de acesso. | Ao logar com um usuário de perfil administrativo, este, ao ser redirecionado à página *Home*, deve ser capaz de visualizar os botões de cadastar usuário e cadastrar novo projeto além das views comuns a todos os usuários, como o botão de Visão Geral, projetos e atividades em andamento. | - Acessar a aplicação *web*.  - Informar usuário e senha de usuário com perfil administrativo, sendo equivalante ao perfil de gerente no sistema:  - **usuário**: gerente1  - **senha**: Gerente@1234 | - O usuário é redirecionado à página *Home* com as funcionalidades sistêmicas disponíveis aos administradores na página Home, a saber, cadastrar usuário e cadastrar projeto. |
|  | *Login* na aplicação com controle de perfil de acesso. | Ao logar com um usuário de perfil colaborativo, este, ao ser redirecionado à página *Home*, não pode visualizar opções reservadas ao perfil administrativo, apenas as de caráter comum como as atividades em andamento do usuário logado e os projetos em que ele está envolvido. | - Acessar a aplicação *web*.  - Informar usuário e senha de usuário com perfil de colaborador:  - **usuário**: guilherme  - **senha**: Guilherme@1234 | - O usuário é redirecionado à página *Home* com as funcionalidades sistêmicas disponíveis aos colaboradores, como visão geral, projetos em que estou envolvido e minhas atividades em andamento. |
|  | *Login* na aplicação com controle de perfil de acesso. | Ao logar com um usuário não cadastrado no sistema, este deve retornar um alerta de erro: “Senha ou usuário inválido”, de modo a não revelar a um possível *cracker* qual das duas informações requeridas foi a incorreta. | - Acessar a aplicação *web*.  - Informar usuário e senha de usuário inexistente no sistema:  - **usuário**: usuario-nao-cadastrado  **senha**: senhaqualquer | - Um alerta é realizado na tela de *login* dizendo que usuário ou senha estão incorretos. |
|  | Criação de projeto. | Ao criar um projeto no sistema de gerenciamento, um gerente cria um projeto inputando as informações obrigatórias requeridas e o projeto é criado com sucesso. | - Acessar a aplicação *web*.  - Acessar o sistema utilizando as credenciais administrativas do usuário gerente 1.  - Clicar/Acessar no botão Novo Projeto.  - Após redirecionamento para página de formulário de criação de projeto, realizar as seguintes entradas:  - **Identificação do projeto**: Projeto ABC  - **Gerente do projeto**: gerente 1  - **Descrição do projeto**: texto descritivo do projeto.  - **Data de início**: 14/08/2021  - **Data de término**: 15/10/2022 | - Com as informações obrigatórias preenchidas, a persistência será realizada e o usuário será redirecionado à página *Home* com um alerta dizendo: “Projeto criado com sucesso!”. |
|  | Criação de projeto. | Ao criar um projeto no sistema de gerenciamento, um gerente cria um projeto não inputando todas as informações obrigatórias requeridas e o sistema retorna um alerta para o usuário. | - Acessar a aplicação *web*.  - Acessar o sistema utilizando as credenciais administrativas do usuário gerente 1.  - Clicar/Acessar no botão Novo Projeto.  - Após redirecionamento para página de formulário de criação de projeto, realizar as seguintes entradas:  - **Identificação do projeto**: Projeto ABC  - **Gerente do projeto**: nulo  - **Colaboradores**:  guilherme  - **Descrição do projeto**: texto descritivo do projeto.  - **Data de início**: 14/08/2021  - **Data de término**: nulo | - O formulário requer que identificação do projeto, gerente e data fim sejam preenchidas. Caso alguma destas informações não venha preenchida, como é o caso deste teste, ainda na página de formulário de criação de projeto, o usuário receberá o seguinte alerta:  “Gerente e data fim do projeto devem ser providenciados!” e espera-se que o usuário as preencha para que o projeto seja criado. |
|  | Atualização de tarefa. | Um usuário de perfil colaborador, ao visualizar uma tarefa, quer atualizá-la. A lista de requisitos funcionais diz que o responsável por esta pode atualizá-la e interpretou-se que um usuário de perfil gerencial também pode ter acesso ao botão de atualização. | - Acessar a aplicação *web.*  - Acessar o sistema usando as credenciais de colaborador do usuário guilherme.  - Na página *Home*, clicar/acessar a atividade 1.  - Após o redirecionamento para a página de detalhe da atividade, verificar que o botão ‘concluir’ está ativado e clicar nele. | A atividade deve ser encerrada(concluída) e o usuário deve ser redirecionado à página *Home*. |
|  | Atualização de tarefa. | Um usuário de perfil colaborador, ao visualizar uma tarefa, quer atualizá-la. A lista de requisitos funcionais diz que o responsável por esta pode atualizá-la e interpretou-se que um usuário de perfil gerencial também pode ter acesso ao botão de atualização. Neste caso, um usuário que não é dono da atividade tentará concluir a atividade atribuída a outro colaborador. | - Acessar a aplicação *web.*  - Acessar o sistema usando as credenciais de colaborador do usuário João.  - Na página *Home*, clicar/acessar o projeto ABC.  - Após o redirecionamento para a página de detalhe do projeto, clicar na atividade 1.  - Após redirecionamento para detalhes da atividade 1, tentar realizar sua conclusão. | O usuário João consegue visualizar a atividade 1, tanto as informações relacionadas à descrição da atividade, responsável, datas de início e fim, mas o botão de conclusão náo é habilitado para o usuário colaborador joao, pois a tarefa pertence ao usuário colaborador guilherme. |

## Estimativa de pontos de função

Indique o *link* do repositório (como o [GitHub](http://github.com), por exemplo) onde sua planilha de estimativa de pontos de função está disponível.> + planilha física dos casos de uso no tcc. sad

## Informações da implementação

Seção de implementação

-

Seção de implantação

-

Link para github

## Referências

<Esse trabalho não requer revisão bibliográfica e, por isso, a inclusão das referências não é obrigatória, embora seja recomendada. Caso você deseje incluir referências empregadas em seu trabalho, relacione-as de acordo com as normas ABNT, disponíveis em [www.pucminas.br](http://www.pucminas.br), no *link*: <http://portal.pucminas.br/imagedb/documento/DOC_DSC_NOME_ARQUI20160217102425-n.pdf>.

Exemplo:

SOBRENOME DO AUTOR, Nome do autor. **Título do livro ou artigo.** Cidade: Editora, ano.>